LOCKING THE WEB OPEN:

A CALL FOR A DECENTRALIZED WEB

BREWSTER KAHLE, INTERNET ARCHIVE

JUNE 2016, DECENTRALIZED WEB SUMMIT

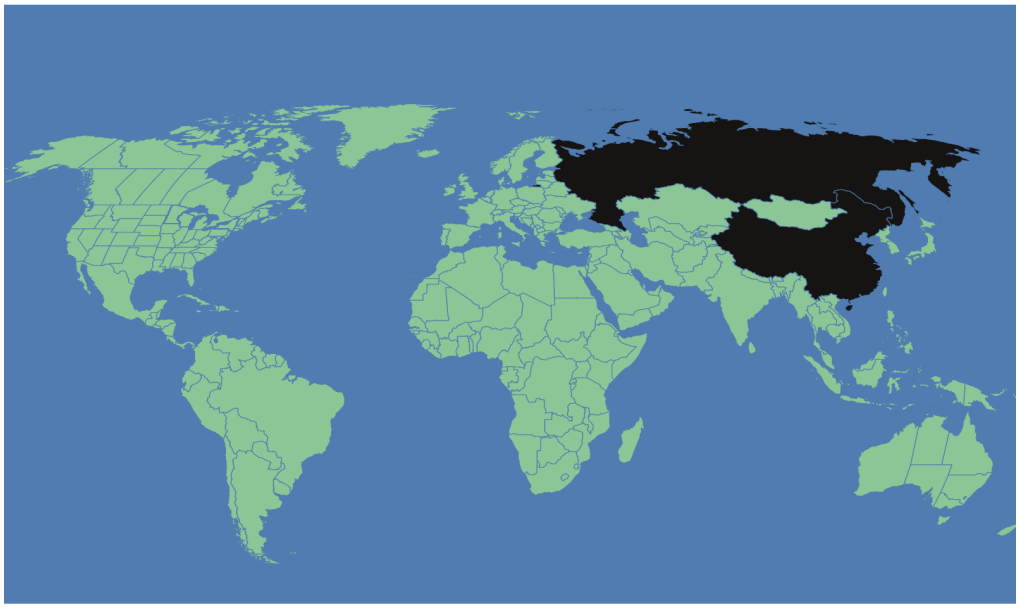
Locking the Web Open:
A Call for a Decentralized Web

Brewster Kahle, Internet Archive
June 2016, Decentralized Web Summit

Over the last 25 years, millions of people have poured creativity and knowledge into the World Wide Web. New features have been added and dramatic flaws have emerged based on the original simple design. I would like to suggest we could now build a new Web on top of the existing Web that secures what we want most out of an expressive communication tool without giving up its inclusiveness. I believe we can do something quite counter-intuitive: We can lock the Web open.

One of my heroes, Larry Lessig, famously said: "Code is Law." The way we code the web will determine the way we live online. So we need to bake our values into our code. Freedom of expression needs to be baked into our code. Privacy should be baked into our code. Universal access to all knowledge. But right now, those values are not embedded in the Web.

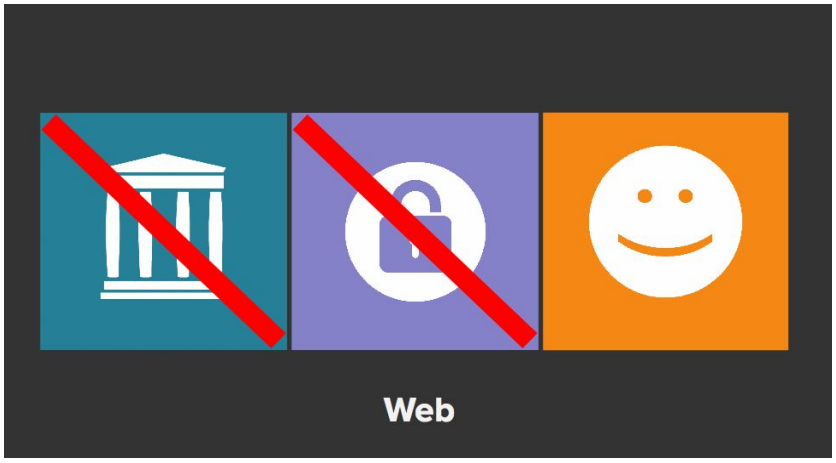
It turns out that the World Wide Web is quite fragile. But it is huge. At the Internet Archive, we collect one billion pages a week. We now know that webpages only last about 100 days on average before they change or disappear. They blink on and off in their servers.



And the Web is massively accessible—unless you live in China or Russia. The Chinese government, and sometimes the Russian government, has blocked the Internet Archive, the New York Times, and other sites from its citizens. And other countries block their citizens' access every once in a while, as well. So the Web is not reliably accessible.

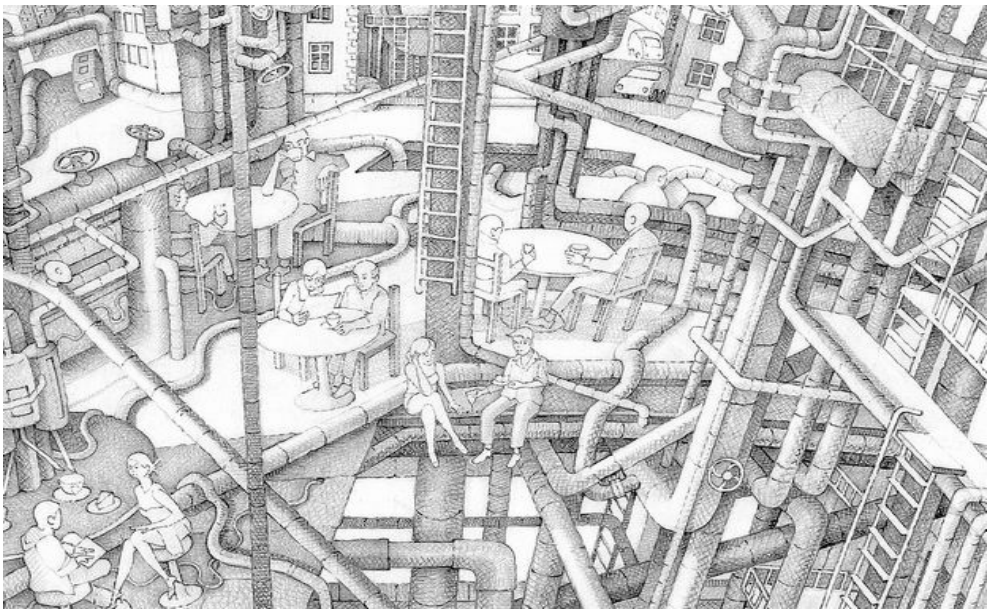
And the Web isn't private. People, corporations, countries can spy on what you are reading. And they do. We now know, thanks to Edward Snowden, that Wikileaks readers were selected for targeting by the National Security Agency and the UK's equivalent just because those organizations could identify those Web browsers that visited the site and identify the people likely to be using those browsers. In the library world, we know how important it is to protect reader privacy. Rounding people up for the things that they've read has a long and dreadful history. So we need a Web that is better than it is now in order to protect reader privacy.

But the Web is fun. The Web is so easy to use and inviting that millions of people are putting interesting things online, in many ways pouring a digital representation of their lives into the Web. New features are being invented and added into the technology because one does not need permission to create in this system. All in all, the openness of the Web has led to the participation of many.



We got <u>one</u> of the three things right. But we need a Web that is reliable, a Web that is private, while keeping the Web fun. I believe it is time to take that next step: I believe we can now build a Web that is reliable, private and fun all at the same time. To get these features, we need to build a "Decentralized Web."

Imagine “Decentralized Web” sites that are as easy to setup and use as WordPress blogs, Wikimedia sites, or even Facebook pages, but have these properties. But how? First, a bit about what is meant by a “decentralized system.”



Contrast the current Web to the Internet—the network of pipes on top of which the World Wide Web sits. The Internet was designed so that if any one piece goes out, it will still function. If some of the routers that sort and transmit packets are knocked out, then the system is designed to automatically reroute the packets through the working parts of the system. While it is possible to knock out so much that you create a chokepoint in the

Internet fabric, for most circumstances it is designed to survive hardware faults and slowdowns. Therefore, the Internet can be described as a “decentralized system” because it routes around problems and automatically rebalances loads.

The Web is not decentralized in this way. While different websites are located all over the world, in most cases, any particular website has only one physical location. Therefore, if the hardware in that particular location is down, then no one can see that website. In this way, the Web is centralized: if someone controls the hardware of a website or the communication line to a website, then they control all the uses of that website.

In this way, the Internet is a truly decentralized system, while the Web is not.

Decentralized systems are typically more difficult to design than centralized ones. At a recent talk by Vint Cerf, sponsored by the California Academy of Sciences, Cerf said that he spent much of 1974 in an office with two other engineers working on the protocols to support a decentralized Internet system, to make it such that there are no central points of control.

Here's another way of thinking about decentralized systems: take the Amazon Cloud. The Amazon Cloud is made up of computers in Amazon.com data centers all over the world. The data stored in this cloud can be copied from computer to computer in these different places, avoiding machines that are not working, as well as getting the data closer to users and replicating it as it is increasingly used. This has turned out to be a great idea. What if we could make the next generation Web work like that, but across the entire Internet, like an enormous Amazon Cloud?

In part, it would be based on peer-to-peer technology—a system that isn't dependent on a central host or the policies of one particular country. In a peer-to-peer model, those who are using the decentralized Web are also providing some of the bandwidth and storage to run it.

Instead of one Web server per website, we would have many. The more people or organizations that are involved in the decentralized Web, the more redundant, safe, and fast it will become.

And it also needs to be private—so no one knows what you are reading. The bits will be decentralized—across the net—so no one can track the readers of a site from a single point or connection. Absolute privacy may be difficult to achieve, but we can make the next Web much more secure.

The next generation Web also needs **a decentralized authentication system without centralized usernames and passwords**. That's where cryptography comes in to provide a robust but private identity system.

We'd also want to bring in some other features if we're going to redo this Web.

This time, the **Web should have a memory**. We would like to build in a form of versioning, so the Web is archived through time. The Web would no longer exist in a land of the perpetual present.

On library shelves, we have past editions of books, but on the Web, you don't have past editions of websites. Every day is a new day, unless you know to use the Internet Archive's Wayback Machine, which may have copies of previous versions. Where the Wayback Machine was created after-the-fact to solve this problem of the current Web, in this next iteration we can build versions into the basic fabric of the Decentralized Web to provide a history and reliability to our growing digital heritage.

We could also add a feature that has long been missing from the Web: **easy mechanisms for readers to pay writers**. With the support of easy payments, the Decentralized Web could evolve richer business models than the current advertising and large-scale e-commerce systems.

Adding redundancy based on decentralized copies, storing versions, and a payment system could reinforce the reliability and longevity of a new Web infrastructure.

Plus it needs to be fun—malleable enough to spur the imaginations of millions of inventors. This new Web could be an inviting system that welcomes people to share their stories and ideas, as well as be a technology platform that one can add to and change without having to ask permission—allowing technological change just for the fun of it.

How can we build this new Decentralized Web? There have been many advances since the start of the Web in 1992 that will be helpful.

We have computers that are a thousand times faster. We have JavaScript that allows us to run sophisticated code in the browser. So now, many more people can help to build it.

Public key encryption systems were illegal to distribute in the early 90s, but are now legal, so we can use them for authentication and privacy. With strong cryptography, communications can be made safe in transit and can be signed so that forgery is much more difficult.

We have blockchain technology that enables the Bitcoin community to have a global database with no central point of control.

And we have virtual currencies such as bitcoin, which could make micropayments work in a decentralized environment. Many other projects have pushed the limits of decentralized systems giving us building blocks for a Decentralized Web.

I've seen each of the necessary pieces work independently, but never pulled together into a new Web.

I suggest we need a bold goal, one that is understandable and achievable. Something that we might be able to rally around, and have multiple groups contribute to, in order to build an easy to use Decentralized Web.

What about Wordpress, but decentralized? Wordpress is a very popular toolkit that millions have used to build websites. My blog, brewster.kahle.org, for instance, is built on the open source Wordpress software installed on a server at the Internet Archive. Free to use, and free to host, this toolkit enables anyone to select from a set of template designs and modify it to give it a unique look. Then the original creator can appoint users to play roles such as administrator, editor, or commenter. Those with these different privileges can, in turn, grant privileges to others as appropriate. And then the writers can post articles or images to its pages or change the look and feel of the site.

A Wordpress website, traditionally, would then be hosted on a computer of the creator's choice, either on Wordpress.com, or on other sites offering hosting, or even on their own

computer because the underlying software is available open source as well. This is where Wordpress is not "decentralized," in the sense we were talking about earlier. If the organization hosting the site does not like the material, or it is blocked in another country or goes out of business, then the website will not be available. Even major companies, such as Apple, Google, and Yahoo, have taken down whole systems hosting millions of users' websites, often with little notice.

We would like to allow anyone to build a Wordpress website—that has themes and different people with different roles, fun to go to and add to, free to create—which is also decentralized in a way that is private and reliable.

We would want it to work in all browsers with no add-ons or modifications. We would want to refer to a decentralized website with a simple name like brewster.kahle.org and it needs to be fast.

We would need users to be able to log in without having to have many websites know their usernames and passwords, or have a central site, like Facebook or Google, control their online credentials. In other words, we need a decentralized identity system.

Additionally, we would like to have payments work in the Decentralized Web. We would like to enable anyone to pay anyone else, akin to leaving a tip, or paying a suggested amount for reading an article or watching a movie. Thus, people could get paid for publishing on this Decentralized Web.

Finally, we would want to have saved versions of websites, and dependable archives to make these decentralized websites reliable.

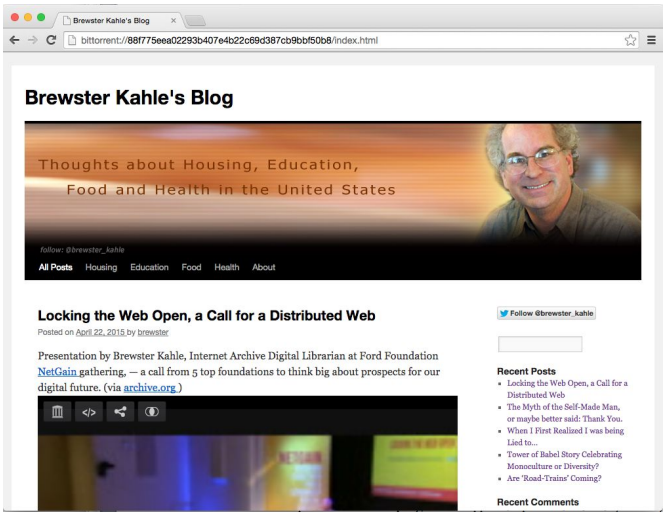
How can we build this system?

Way to Build the Decentralized Web: an Example

Please bear with me as I to try to argue that this is possible using an amalgam of existing or near-existing technologies.

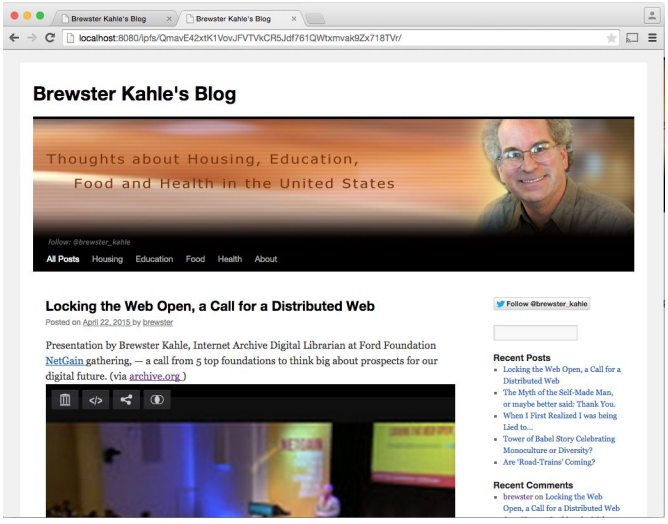
A piece of this system could be a peer-to-peer system such as BitTorrent. Storing and retrieving files in a decentralized way has been commonplace for years with BitTorrent. While downloading custom software is not ideal, it shows this function can be done and done for millions of people. BitTorrent is kind of magic, where typing a long number that is a unique identifier for a file or set of files will cause it to appear on your machine. Pieces of the desired file will come from other computers that had previously retrieved those files and, therefore, store them on their computers. In this way, the readers of files become the servers of those files. There are millions of users of BitTorrent sharing everything from commercial movies, to free software, to library materials. The Internet Archive, for instance, offers petabytes of files to the public using the BitTorrent protocol so that users have the option to retrieve files from the Internet Archive or from other users who might be closer.

Using BitTorrent as part of the Decentralized Web to share the files is working in prototype form now. BitTorrent Incorporated's peer-to-peer powered Web browser Maelstrom is now in alpha release. With this browser, a file of files can be distributed using BitTorrent. Using this early version, I demonstrated at a conference last month a static version of my blog, brewster.kahle.org being served by people around the Internet.



Notice in this image, that the Web address starts with bittorrent:// and then a long number. This is how the website was retrieved from the BitTorrent network.

Another system, IPFS, designed and implemented Juan Benet, is open source and has some of the same peer-to-peer characteristics but has some added enhancements. Juan took my blog and in a few minutes put it into his system, showing that system is also working. One of the major features this system offers over BitTorrent is that updates to the blog can be discovered and distributed naturally through the system. Therefore, as people would add comments and posts to a blog, these can be retrieved without having to get a new identifier.



Notice, in this case, the Web address refers to the localhost, meaning that it is retrieving the pages using a computer program running on my laptop, which is operating the peer-to-peer functionality.

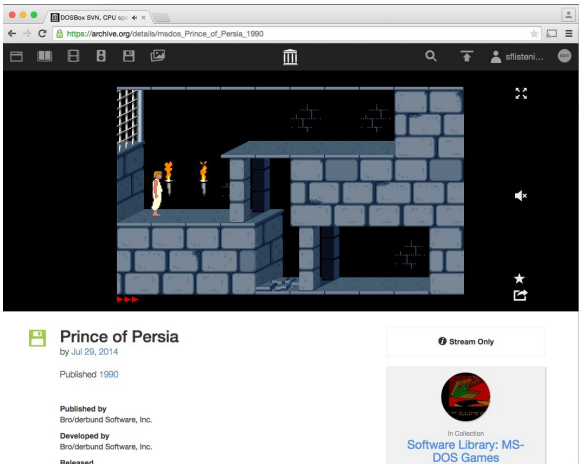
Other decentralized systems are in different stages of development, which will certainly be useful. Many of these systems are listed at the end of this paper.

Therefore, the idea of storing and retrieving files that are part of a decentralized website is now a reality in prototype form. But there are still some pieces missing.

Building Seamlessly on Top of the Existing Web

One way that we can create a bridge between the traditional Web and the decentralized Web would be to have decentralized websites that work seamlessly in readers' browsers without any add-ons, plug-ins, or downloads—just click and see.

This is important because software on phones, tablets, and laptops are becoming more difficult to install without the permission of a company, such as Apple. Fortunately, it is easy to distribute JavaScript as part of webpages, and this will likely be supported for a long time because it is important to sites such as Google Docs and Google Maps.



JavaScript running in users' browsers as a kind of application platform is now possible and usable. I was surprised to find that JavaScript is now powerful enough to emulate older computers in the browser. For instance, you can now run an IBM PC emulator running MSDos 3.1 running a computer game just by clicking on a weblink to go to a webpage. The

game Oregon Trail, or Prince of Persia, or old arcade games are now available on the Internet Archive and have been played by millions of people. The way this works is that others have made emulators of the underlying machines in the programming language C, and then that code cross-compiles it into JavaScript. So, when a user goes to Archive.org and clicks to run it, it downloads a JavaScript program that boots an emulator of an old IBM PC or an Apple 2 in the browser. Then it reads a floppy, in this case a virtual floppy, and then runs that program in the emulator so that you are basically experiencing that old computer interface. It was a strange mind twist for me to download and run a whole machine emulator in a browser. Since JavaScript is capable enough to do that, then we can build the mechanism we need for the Decentralized Web in JavaScript.

To run a decentralized system in the browser, we need one more feature. The code running in the browser must be able to connect to other browsers that are running the same system. Basically, we need to make it so that a browser can contact another browser instead of going to a server. This is now achievable based on a new standard, WebRTC, that was created to allow video conferencing and multiplayer games.

With the underlying speed of modern machines, the maturity of a coding system like JavaScript, and the peer-to-peer features supported in browsers, we seem to have all the pieces we need to support a Decentralized Web on top of the current Web without any downloads, plug-ins, or add-ons.

There is an additional advantage to building the Decentralized Web in JavaScript: it can be changed and added to by many people independently. In fact, different websites might use different Decentralized Web systems all interoperably on the Internet at one time. It does not require coordination or relationships with the browser manufacturers to make changes to how the Decentralized Web works. Features can be added, subtracted, and experimented with in parallel, without permission. The Decentralized Web could evolve much faster than current Web technologies and yet still be interoperable.

Decentralized Websites that have Search Engines and Databases

Since Wordpress sites have search and database functions for selecting posts from particular months and with particular tags, to be fully functional, we need our decentralized websites to have these features as well. In the current Web, programs running on a server support these features, so that when the user types a few words into a search box, it is sent to the server, and then a program runs on the server to create the page that is then transmitted back to the browser. In the Decentralized Web there are no servers, there are only static files that are retrieved from a peer-to-peer network. Luckily, some of the files of the website can themselves be computer code in the form of JavaScript. All of the computation then happens in the browser based on those files.
Fortunately, this is possible because a search engine and the index can be packaged as files that can be downloaded to a browser and run in the browser. This feature has been achieved in the demonstrations based on BitTorrent as well as IPFS mentioned before; the programmer, Marcel van der Peijl, used an open source tool js-search to take the pages of my blog site to create an index plus search engine in JavaScript. For my site, the resulting JavaScript page was one megabyte, which is large, but not too large for broadband computer users. To make this more usable, he only loads this code after the page the user requested has been displayed, so in most cases the user would not notice the delay.

This approach will work for most blogs, but maybe the largest ones will need more sophistication.

Therefore, we can have decentralized websites that include dynamic elements such as search engines and databases.

Adding New Posts and Other Changes to a Decentralized Website

A key feature of a Wordpress site is adding comments or posts. This is trickier in a decentralized setting than in centralized systems because updates have to be made in many places. In the Wordpress application we do not need the website to be up-to-the-second for every reader, but we need to propagate changes quickly.

BitTorrent has a facility called “mutable torrents” which allows updates, but currently this requires a centralized facility to keep track of the newest version. This has the disadvantage of making the user contact a central server to find the most up-to-date version. This central server could be watched or controlled by a third party.

Another peer-to-peer file sharing system, IPFS, on the other hand, has a truly decentralized facility for supporting updates and versions. IPFS is a very clever system that has some of the features of a Unix file system, but one that supports versions. How this works is some of the genius of this system. Since we have seen that a decentralized Wordpress site can be made out of files, which may contain images and text, as well as code that can then be retrieved and played in the browser, a decentralized file system could hold and transmit required files.

So there are solutions, even in a decentralized way, to have millions of updates and not have to resort to central control or central notification that could impact our goal of protecting reader privacy.

The Wayback Machine of the Decentralized Web

The Wayback Machine is a free service of the Internet Archive that allows people to see past versions of websites. We do this by periodically visiting a website and downloading and storing the webpages being offered.

A Wayback Machine for the Decentralized Web could store versions as they are created because it is easy to recognize what has changed and store it. This way, the future Wayback Machine would have some significant advantages over the current one—it could always be up to date and it could help serve the current website and past versions to users without their even knowing it. This way the user would not need to go to the archive.org website to access the Wayback Machine. It would just start serving versions of the website on request, including the current version. If it did not have those files, then it could find them from other servers to add to the archive. Therefore, the Wayback Machine would be a form of host for the current version of the website, since it would participate in offering files to the readers. The Wayback Machine would, therefore, make the Decentralized Web more reliable.

If someone referred to a past version of a website, and if the Wayback Machine had those files, it would serve those as well. In this way, the Wayback Machine would become more tightly integrated into the Decentralized Web.

Many Wayback Machines could be run by many different organizations in a smooth way. As more groups participate, the more reliable and robust this system would become.

There is another significant advantage to the Wayback Machine application in the Decentralized Web: it would archive and serve fully functional websites, not just snapshots of what it looked like through time. All of the functionality would be served, so its search

and database functions would be supportable forever and in past versions. This way, the decentralized websites would live on in time and space even if there were a disruption in hosting or authorship.

In this way, a library, such as the Internet Archive, could preserve and provide access to websites that are no longer maintained, or where the authors have moved on to other projects. This is similar to what libraries have done with professor's research papers—offering enduring access to past works so that people can learn from them.

Therefore, the Decentralized Web would have a major advantage because it could be easily archived and served in a decentralized and enduring way.

Fast performance

By having institutions such as the Internet Archive offering access to decentralized websites, the users will get a more reliable service, but it could also help provide better performance. Since there are other organizations that are also motivated to provide fast and reliable access for their users, others could help replicate the data and make the Decentralized Web more robust. Internet Service Providers (ISPs), for example, want their users to have a good Web experience and would be likely to serve as a close and fast host for their users. This would also help save those companies on bandwidth bills because more of their traffic would be local. In this way, there can be cultural institutions as well as commercial organizations that have incentive to replicate parts of the Decentralized Web, thus increasing reliability and performance for users.

Surveillance and Censorship

Since the Decentralized Web would have users and repositories all over the world, both hosted by institutions and by other readers of the Decentralized Web, some of the techniques for surveillance and censorship would become more difficult. For instance, the so-called Great Firewall of China blocks access to some websites outside of China by watching all traffic on its borders and filtering based on which websites are being accessed. Since a decentralized website does not have a single location it would be more difficult to monitor or block its use. Furthermore, if one copy gets behind a firewall of this kind, then it can be replicated inside, making censorship more difficult.

The encryption used in this traffic may make it difficult to even know which files are being requested in the first place. Therefore, some of the existing systems of surveillance and censorship will not be as easy to use in the Decentralized Web.

blockstack

Easy Names of Decentralized Websites

We also want easy-to-remember names for decentralized websites. When the Internet was first designed, there were IP addresses that were strings of numbers such as 207.241.224.2. These were not easy to remember so a naming system was created called the Domain Name System (DNS), which allowed someone to remember names such as "archive.org" instead of an address. The Web, being built on the Internet, used these in its universal resource locators, such as http://archive.org.

In the Decentralized Web, we have a similar problem with long, hard-to-remember numbers. In the implementations described above for both BitTorrent and IPFS, a webpage is a unique, incomprehensible string such as: 88f775eea02293b407e4b22c69d387cb9bbf50b8 or /ipfs/QmavE42xtK1VovJFVTVkCR5Jdf761QWtxmvak9Zx718TVr. It would be much more convenient if we had a string such as https://brewstersblog.arc.

The domain name service could be used for this purpose and would probably be a good starting point because it would leverage a large investment in technology and investments by society in regulating who gets what names. The Decentralized Web could also incorporate new naming systems that would exist alongside the DNS to support new approaches to naming and the technologies to support them.

One decentralized naming system that currently exists is called Namecoin, and it is an open source system built on a Bitcoin-like blockchain, which is in itself a decentralized system.

To understand Namecoin, let's start with some of the characteristics of blockchain technology.

A blockchain is a form of decentralized database that is used to store the ledger in Bitcoin and similar systems. It is very clever in how it maintains consistency even when none of the participants trust each other. People submit "transactions" by signing them with their private cryptographic keys, and offer a financial tip to those who compete to operate the blockchain consensus system; these are the so-called "miners." The blockchain is then a way to record transactions that everyone can see and everyone agrees to. In the case of Namecoin, the blockchain is used to register a claim for a name and the long number with which it will be associated.

In this way, people can register a name and address pair in the blockchain and others can look it up in a decentralized manner. Unfortunately, looking up a name is a time-consuming process, but at least it is certain who registered a name first. Increasing performance can be another task.

Another system that could be used for this is the Decentralized Hash Table, or DHT, which is central to the way BitTorrent works. This is another decentralized system for looking up a name.

So if this is done correctly, we can have easy-to-remember names resolve to decentralized websites quickly, securely, and privately.

Furthermore, there could be registrars that charge for new names, and in return, offer services such as fast servers and permanent archives. This could be a new business model that helps support the system.

To have a decentralized naming system work in current browsers, without modification, we need a way to resolve the name to an address in JavaScript without contacting the server. Fortunately, there is a mechanism to do this using an anchor tag such as https://dweb.org/#brewstersblog.arc.

Therefore, we can have a simple system for naming decentralized websites without losing privacy or reliability.

Decentralized Identity

To know who is allowed to update a blog, we need a system to register administrators and then to authenticate someone as being that person. That is achieved on current Wordpress sites when a user creates an account with a username and password using a webpage. This is kept in a database on the server. If a similar system could be implemented with a decentralized webpage that operates the database, we could make the system more secure and easier for people to use.

Another way current websites often work is that one can log in using one's Google, Facebook, or Twitter account information. This way a user does not have to give a password to many different sites, but it has the disadvantage that large corporations know a great deal about one's behavior online.

A better system might be one that uses cryptography to allow users to create multiple account credentials and use these without necessarily tying them back to their persons. That way people would have control over who knows what about them, and if they wanted to walk away from an account, that would work as well.

This could use what is called public key cryptography, which uses special math functions to create pairs of public and private keys. The private key is used to sign documents in such a way that anyone using the public key, which is publicly known, can verify that it was correctly signed. No one else can forge a document. Thus, if posts were signed on a Decentralized Web, then the readers can verify that it is the particular user that has the authority to perform that action and the website never needs to know a user's password or private keys.

Making Money by Publishing on the Decentralized Web

Public-private key pairs are central to how Bitcoin works, and this fact can be useful. In Bitcoin, a public key is used as the account name such as 1KAHLE1taA85EXaVm1XuVYtbGp839MyEzB. With Bitcoin, people can create as many accounts as they want to. An account really has an effect only when someone has created a transaction using it, thereby depositing bitcoin into that account. Anyone can deposit money (bitcoin) into an account, but only the holder of the private key for an account can transfer the money out of the account and into another account.

If the Decentralized Web uses the same math function for creating public and private keys that Bitcoin does, then the Decentralized Web's identity system will be compatible with Bitcoin accounts. This has an interesting advantage that anyone could leave a tip for any writer on the Decentralized Web because his public key would be his Bitcoin account. In this way, we could make it easy for payments, even very small ones, to be made in the Decentralized Web.

I believe it would even be possible to use Bitcoin-like technology to require a payment before a reader can decode a file, say a movie. In this way, we may have a decentralized way to sell digital files on the Internet without any central clearinghouse. It would still be possible to rip someone off by buying a file, decoding it, and then redistributing it, but this is true now. What would be different is that it would be easy to make micropayments and full purchases on the Decentralized Web without third parties getting involved or taking a slice. Automated tipping could even be installed to try micropayments as a default behavior.

Locking the Web Open

In conclusion, through the last 25 years, people have poured their lives and dreams into the World Wide Web, yielding a library and communication tool that is unprecedented in scale. We can now build a stronger tool on top of the current Web to offer added reliability, privacy, and fun.

Our new Web would be reliable because it would be hosted in many places, and support multiple versions. Also, people could even make money, so there could be extra incentive to publish in the Decentralized Web.

It would be more private because it would be more difficult to monitor who is reading a particular website. Using cryptography for the identity system makes it less related to personal identity, so there is an ability to walk away without being personally targeted.

And it could be as fun as it is malleable and extendable. With no central entities to regulate the evolution of the Decentralized Web, the possibilities are much broader.

Fortunately, the needed technologies are now available in JavaScript, Bitcoin, IPFS/BitTorrent, Namecoin, and others. We do not need to wait for Apple, Microsoft, or Google to allow us to build this.

What we need to do now is bring together technologists, visionaries, and philanthropists to build such a system that has no central points of control. Building this as a truly open project could itself be done in a decentralized way, allowing many people and many projects to participate toward a shared goal of a Decentralized Web.

Together we can lock the Web open.

We can make openness irrevocable.

We can bake the First Amendment into the code itself, for the benefit of all.

We can build this.

We can build it <u>together</u>.

Previous writings on this subject:

https://blog.archive.org/2012/10/22/siteless-website-possible-if-BitTorrent-is-a-fileserver-without-a-server-what-about-a-website-without-a-site/

https://blog.archive.org/2012/02/15/want-to-help-build-a-distributed-web/

https://blog.archive.org/2015/02/11/locking-the-web-open-a-call-for-a-distributed-web/

http://thenextweb.com/insider/2015/04/10/locking-the-web-open-why-we-need-to-rethink-the-world-wide-web/

Decentralized systems and communities: Maelstrom by BitTorrent, MaidSafe, Namecoin, Ethereum, Bitcoin for payments, Blockstack DNS and PKI, Proof of Storage (blockchain), ZeroNet, Oceanstore, I2p, IPFS, Storj, Peer5, Tahoe-LAFS, Twister, Peerjs / Web RTC, BitcoinJS, Redecentralize.org, get-d.net.